PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-040414
(43)Date of publication of application : 13.02.1998

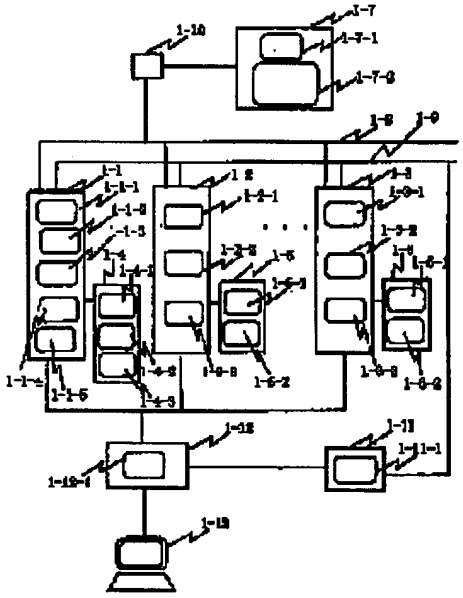
(51)Int.Cl. G06T 15/00
G06F 15/16
G06T 1/20
// G06T 1/60

(21)Application number : 08-210577
(22)Date of filing : 22.07.1996

(71)Applicant : HITACHI LTD
(72)Inventor : SUGITA YUMIKO
KIMURA SHINJI
KUWANA TOSHIYUKI

(54) MULTIPROCESSOR PLOTTING AND PROCESSING DEVICE

(57)Abstract:
PROBLEM TO BE SOLVED: To reduce the time for access in the case of referring to data without being limited by the capacity, data amount or data constitution of respective local memories in the case of ray-casting system calculation.
SOLUTION: A shared memory 1-7 stores three-dimensional numerical data and information concerning the constitution of data and a data reference pattern and a processor 1-1 divides processing and distributes processing to processors from 1-1 to 1-3. The respective processors are parallelly operated, perform processing for finding the arrangement information and buffer sizes of reference data at the time of calculation respectively, secure buffers in local memories from 1-4 to 1-6, store the arrangement information or the buffer sizes in management information tables inside the local memories and afterwards generate two-dimensional images by executing processing partially charged in plotting processing. When the buffers can not be secured because local memories are lacked, it is reported to the processor 101, this processor refers to the working condition information of respective processors, divides processing again and transmits it to the respective processors and based on this transmission, the respective processors execute processing.



LEGAL STATUS

[Date of request for examination] 28.02.2000

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]	3292976
[Date of registration]	05.04.2002
[Number of appeal against examiner's decision of rejection]	
[Date of requesting appeal against examiner's decision of rejection]	
[Date of extinction of right]	

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平10－40414

(43)公開日　平成10年(1998)２月13日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｔ　15/00			Ｇ０６Ｆ　15/72	４５０Ｋ	
Ｇ０６Ｆ　15/16	３８０		15/16	３８０Ｚ	
Ｇ０６Ｔ　1/20			15/66	Ｋ	
// Ｇ０６Ｔ　1/60			15/64	４５０Ｆ	

審査請求　未請求　請求項の数４　ＦＤ　（全 15 頁）

(21)出願番号　特願平8－210577

(22)出願日　平成８年(1996)７月22日

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地

(72)発明者　杉田　由美子
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内

(72)発明者　木村　信二
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内

(72)発明者　桑名　利幸
茨城県日立市大みか町五丁目２番１号　株式会社日立製作所大みか工場内

(74)代理人　弁理士　笹岡　茂　（外１名）

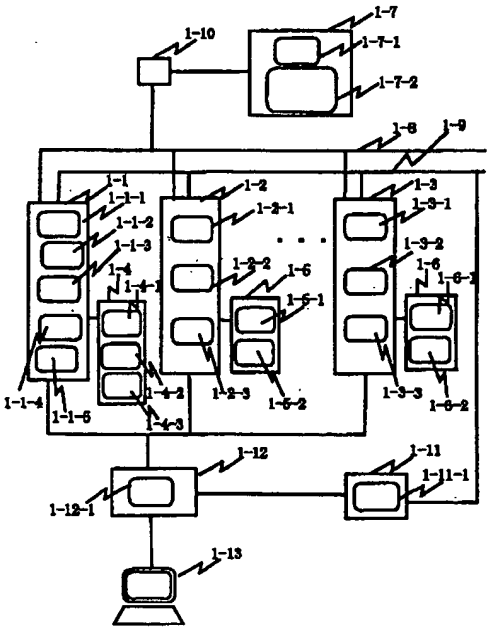
(54)【発明の名称】　マルチプロセッサ描画処理装置

(57)【要約】

【課題】　レイキャスティング方式の計算において、各ローカルメモリの容量やデータ量、データ構成に制限されずにデータ参照時のアクセス時間を減らす。

【解決手段】　共有メモリ1-7には３次元数値データとデータの構成とデータ参照パターンに関する情報が格納され、プロセッサ1-1は処理を分割し、プロセッサ1-1～3に処理の分配を行う。各プロセッサは並列動作し、夫々計算時の参照データの配置情報とバッファサイズを求める処理をし、ローカルメモリ1-4～6内にバッファを確保し、配置情報やバッファサイズはローカルメモリ内の管理情報テーブルに格納し、その後、描画処理で分担した処理を実行して２次元画像を生成する。ローカルメモリ不足でバッファ確保できない場合には、それをプロセッサ1-1に通知し、該プロセッサはプロセッサの稼働状況情報を参照し、処理を再分割して各プロセッサへ送信し、これに基づき各プロセッサは処理を実行する。

図 1



1

【特許請求の範囲】

【請求項１】　大容量の３次元数値データを格納する手段と、複数のプロセッサと、前記数値データ格納手段とは独立に前記プロセッサ個々に割り当てられたローカルメモリとで構成し、大容量３次元数値データを用い、前記複数プロセッサがそれぞれ担当処理範囲の情報に従いレイキャスティング法に基づく計算により２次元画像を生成して可視化する描画処理装置において、

一連の前記計算で参照するデータの配置情報を収集する手段と、

前記数値データ格納手段から読み込んだデータを重ならずに格納するバッファサイズを前記収集する手段を用いて算出し前記ローカルメモリに該バッファサイズのバッファを確保する手段と、

前記計算をする際に参照する前記３次元数値データを前記ローカルメモリのバッファに格納し、２回目以降の参照時にはデータがすでにローカルメモリのバッファに存在するかを識別し、存在すればローカルメモリのバッファから読み込む手段と、を前記各プロセッサが有し、各プロセッサが並列に動作することを特徴とするマルチプロセッサ描画処理装置。

【請求項２】　請求項１記載のマルチプロセッサ描画処理装置において、

前記バッファを確保する手段は、前記ローカルメモリの空き領域サイズを得てその範囲内でバッファサイズを求めることを特徴とするマルチプロセッサ描画処理装置。

【請求項３】　請求項２記載のマルチプロセッサ描画処理装置において、

前記複数のプロセッサのいずれか１台のプロセッサを全プロセッサの稼働状態を管理するプロセッサとし、

各プロセッサは、ローカルメモリにバッファを確保できない場合に前記管理プロセッサに稼働不可能の通知と前記担当処理範囲の情報を伝える通信手段を有し、

通知を受けた前記管理プロセッサは、前記稼働不可能なプロセッサ担当分の処理を稼働可能な他のプロセッサに分配する手段を有することを特徴とするマルチプロセッサ描画処理装置。

【請求項４】　請求項１記載のマルチプロセッサ描画処理装置において、

前記各プロセッサは、

１回目の実行時にレイキャスティング法での各ピクセルに対する計算時に参照した前記３次元数値データの奥行き方向の最大処理幅を得る手段と、

同じデータでの２回目の実行時に該手段で得た値を用いてバッファサイズを再計算を行ない、得たサイズが前回よりも小さい場合にはバッファを再確保する手段を有することを特徴とするマルチプロセッサ描画処理装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は画像及びＣＧ描画処

2

理システムに係わり、特にボリュームレンダリグ処理やレイトレーシング処理など、レイキャスティング方式で３次元数値データから２次元画像を算出する処理に好適なマルチプロセッサ描画処理装置に関する。

【０００２】

【従来の技術】画像の分野では最近、医療分野におけるＭＲＩやＣＴデータを利用したボリュームレンダリング処理や３次元ＣＧにおける高品質機能を実現するレイトレーシング処理が適用されている。ボリュームレンダリングとはモデルの表面を対象とするのではなく、測定結果から得られたデータが表現されている３次元離散データのボクセルが持っている数値データ（温度、密度、強度など）と輝度に応じた色を、視線トレースに沿って２次元のピクセル空間に累積写像して表示するものである。なお、ボクセルとは３次元空間をｘ，ｙ，ｚ方向に小さな単位の格子に分割して構成したもので、対象となるデータがこの格子内にあれば１、無ければ０の値で表現される。ボリュームレンダリングの詳細な手法に関しては「ボリューム・ビジュアライゼーションの基本アルゴリズム」ＰＩＸＥＬ　ＮＯ．１２１　ＰＰ１３０－１３７に記述されている。

【０００３】このボリュームレンダリングにおいて高精度／高品質／高画質な画像を得るためには、１ボクセルを求めるのに複数のデータを参照して正規化や輝度の計算を行なわねばならないため、全ピクセルを計算するには参照するデータは膨大で、処理全体で参照するデータ数が数メガバイトのオーダになることも珍しくない。よりリアルで正確な高品質画像を得ようとすればするほど、参照するデータ量や参照回数は多くなり、その分実行時間がかかる。バスも高速になりメモリも大容量になってはきているが、価格や実装上の構成の問題もあり、まだ低価格では実用的な性能を出すに至っていない。

【０００４】この実行時間を短縮する方法として、以下のような参照するデータの参照回数や参照方法を工夫する技術が考えられている。マルチプロセッサ構成のシステムにおいては、対象のデータをマルチプロセッサ数で分割し各プロセッサのローカルメモリに各々転送、各プロセッサはローカルメモリ内のデータのみの処理をし、他のプロセッサのメモリ内のデータでの処理が生じたらそのデータを持つプロセッサに処理を渡し、少ないメモリ量での処理を行ない並列処理による高速化を行なう方式が特開平６－２７４６４７「ローカルメモリ型並列可視化装置」に示されている。対象のデータ群すべては共有メモリに格納しておき、処理で参照するデータのアクセスを一度トレースし、トレース情報を保管し、２回目からの実行ではこのトレース情報を基に共有メモリからデータをキャッシュに先読みして、メモリアクセスのオーバヘッドを削減し高速化を行なう方式が特開平６－３２４９４２「並列計算機システム」に示されている。しかしレイトレーシングやボリュームレンダリングのよう

に、膨大な量のデータを参照し、３次元空間内の任意の視点からの可視化が要求されるシステムにおいては、これらの方式ではまだ課題が多い。

【０００５】マルチプロセッサでデータを均等に分割分配し負荷を分散させ、各プロセッサでそのデータに対してのみ処理を行なう方式では、各プロセッサに分割後のデータすべてを格納できるサイズ分のメモリがあることが条件であり、任意の１つのプロセッサが他の処理でローカルメモリを用いていた場合など、分割データがローカルメモリに格納しきれない時の事は考えられていない。また、３次元数値データには描画処理が参照しないデータも多くあり、その分のデータ転送は処理のオーバヘッドとなる。さらに他のプロセッサのデータを使用する場合にはプロセスを移す必要があるが、視点角度によっては何度も行なわなくてはならず、通信のオーバヘッドが多くなりかえって時間がかかる可能性がある。また、トレースを用いた方式では、データの内容や表示条件によっては視点角度が違うと参照するデータが異なるため、事前に得たトレース情報が役に立たない場合が多い。さらに常に高速に実行するためにはあらゆる条件に対するトレース情報を収集し管理しておく必要があり、この情報量は無視できない大きさになる。

【０００６】

【発明が解決しようとする課題】膨大なデータの参照処理が多い描画処理を行なうマルチプロセッサシステムにおける課題には、以下の項目がある。第１の課題は、参照データの容量やプロセッサ数、各プロセッサのローカルメモリの容量に制限されること無く、さらには無駄なデータ転送を行なわずに、データのアクセス時間を減らす仕組みをローカルメモリのバッファを用いて実現し、高速に処理を実行することである。第２の課題は、任意のプロセッサがローカルメモリ不足で実行不可能である場合でも、処理を続行できるようにすることである。第３の課題は、一度行なった描画処理から参照データのトレースなど大量の情報を収集するのではなく、少ない情報量で処理に適したバッファサイズを得て最適化と高速化を図ることである。

【０００７】

【課題を解決するための手段】上記課題を実現するため、本発明は、大容量の３次元数値データを格納する手段と、複数のプロセッサと、前記数値データ格納手段とは独立に前記プロセッサ個々に割り当てられたローカルメモリとで構成し、大容量３次元数値データを用い、前記複数プロセッサがそれぞれ担当処理範囲の情報に従いレイキャスティング法に基づく計算により２次元画像を生成して可視化する描画処理装置において、一連の前記計算で参照するデータの配置情報を収集する手段と、前記数値データ格納手段から読み込んだデータを重ならずに格納するバッファサイズを前記収集する手段を用いて算出し前記ローカルメモリに該バッファサイズのバッファを確保する手段と、前記計算をする際に参照する前記３次元数値データを前記ローカルメモリのバッファに格納し、２回目以降の参照時にはデータがすでにローカルメモリのバッファに存在するかを識別し、存在すればローカルメモリから読み込む手段と、を前記各プロセッサが有し、各プロセッサが並列に動作するようにしている。

【０００８】さらに、前記バッファを確保する手段は、前記ローカルメモリの空き領域サイズを得てその範囲内でバッファサイズを求めるようにしている。

【０００９】また、前記複数のプロセッサのいずれか１台のプロセッサを全プロセッサの稼働状態を管理するプロセッサとし、各プロセッサは、ローカルメモリにバッファを確保できない場合に前記管理プロセッサに稼働不可能の通知と前記担当処理範囲の情報を伝える通信手段を有し、通知を受けた前記管理プロセッサは、前記稼働不可能なプロセッサ担当分の処理を稼働可能な他のプロセッサに分配する手段を有するようにしている。

【００１０】また、前記各プロセッサは、１回目の実行時にレイキャスティング法での各ピクセルに対する計算時に参照した前記３次元数値データの奥行き方向の最大処理幅を得る手段と、同じデータでの２回目の実行時に該手段で得た値を用いてバッファサイズを再計算を行ない、得たサイズが前回よりも小さい場合にはバッファを再確保する手段を有するようにしている。

【００１１】

【発明の実施の形態】以下、本発明の実施の形態の一例を詳細に説明する。図１は、本発明を適用するシステムの構成図である。ルートプロセッサ１-１はユーザからの描画要求を受け付けると、３次元数値データ１-７-１とデータの構成とデータ参照パターンに関する情報１-７-２を共有メモリ１-７へ格納し、描画条件の取得し、処理の分配を行なう機能１-１-５、を実行する。分割した処理に関する情報は、ルートプロセッサからバス１-９を経由して複数のプロセッサ群１-２～３にデータ転送する。複数のプロセッサ群１-２～３は分割された処理を各々分担し並列に動作する。各プロセッサ１-１～３は共有メモリ１-７からバス１-８を経由してデータの入出力を行なう。この時、複数のプロセッサからの読み出し／書き込み要求はバスブリッジ１-１０で制御する。各プロセッサ内では描画処理に先立ち、計算時の参照データの配置情報とバッファサイズを求める処理１-１-１、１-２-１、１-３-１を実行し、ローカルメモリ１-４～６内にバッファ１-４-１、１-５-１、１-６-１を確保する。配置情報やバッファサイズはローカルメモリ１-４～６内にある管理情報テーブル１-４-２、１-５-２、１-６-２に格納する。その後、描画処理１-１-２、１-２-２、１-３-２で分担した処理を実行して２次元画像を生成する。この描画処理の中でデータの視点トレース回数の算出を行ない、ローカルメモリ１-４～６

内にある管理情報テーブル１-４-２、１-５-２、１-６-２に設定する。なおローカルメモリ１-４～６は共有メモリ１-７とは独立に各プロセッサと接続されている。

【００１２】バッファ確保処理時、ローカルメモリが不足して確保できない場合には、他プロセッサとの通信処理１-１-３、１-２-３、１-３-３を用いて、ルートプロセッサ１-１に通知する。ルートプロセッサは任意プロセッサからの通信を受け取ると、接続されているプロセッサの稼働状況情報１-４-３を参照し、処理を再分割して任意複数プロセッサへ分配送信する処理１-１-４を実行する。各プロセッサ１-１～３で生成した２次元画像の値１-１１-１はフレームバッファ１-１１へ格納し、表示処理プロセッサ１-１２へ渡った表示命令を表示処理機能１-１２-１で処理した時に、画面１-１３へ表示する。なお、本実施例では２次元画像を生成する描画計算はボリュームレンダリングを適用する。

【００１３】図２は、本発明の要素として各プロセッサが得る計算時の参照データの配置情報とバッファサイズ情報１－４－２、１－５－２、１－６－２の詳細を示したものである。各プロセッサが得る情報には、少なくとも以下の情報を持つ。まず共有メモリ上にある３次元数値データに関して、その先頭アドレス２－１と、３次元データ個々のデータ長２－２と、３次元データ全体のｘ方向サイズ２－３、３次元データ全体のｙ方向サイズ２－４、３次元データ全体のｚ方向サイズ２－５、３次元データのデータ構成識別子２－６を持つ。データ構成識別子については図３で説明する。情報２－２～６は３次元データの構成情報であり、ルートプロセッサ１－１が任意の媒体から３次元数値データを共有メモリに読み込んだ時に識別し、他プロセッサにも通知する。その他には得た情報を共有メモリに格納し各プロセッサが読みに行って得る方法もある。これらの情報はボリュームレンダリング処理でデータの参照位置を制御するのに用いる。本実施例ではバッファサイズの計算材料としても用いる。

【００１４】各プロセッサに割り当てられた描画対象領域の情報として、各プロセッサの処理担当領域の個数２－７と、その領域の開始位置のｘ座標２－８、２－１０、開始位置のｙ座標２－９、２－１１を持つ。これらは並列処理を行なうための情報であり、各プロセッサの処理担当部分は１以上の領域から成り、それらの位置を示すすべてのｘ座標とｙ座標の情報を持つ。分割はルートプロセッサが行ない、描画命令と一緒に通知されてきたこれらの情報を各プロセッサの管理情報テーブルに格納する。バッファサイズ計算用のバッファのｘ方向のサイズ２－１２、ｙ方向サイズ２－１３、ｚ方向サイズ２－１４は、計算時に参照するデータの配置情報から得る。これは、全プロセッサで行なう。その他にも、ルートプロセッサが代表して行ない各プロセッサに通知する方法がある。参照データの配置情報とバッファの方向サイズと全体サイズの求め方についてはバッファの管理方法とともに図４～８で説明する。

【００１５】視線トレースの最大回数、すなわちｚ方向の参照幅の最大値２－１５は描画実行時に算出して格納する。さらにローカルメモリに確保したバッファの先頭アドレス２－１６と、そのバッファサイズ２－１７と、バッファのｘ方向サイズ２－１８、バッファのｙ方向サイズ２－１９、バッファのｚ方向サイズ２－２０の情報も持つ。バッファの先頭アドレス２－１６はバッファのＩ／Ｏ制御時に参照する。これらの情報（２－１７～２０）は２回目以降に最適なバッファサイズを再計算する際に使用する。

【００１６】図３は３次元数値データの構成例である。図３に示した３次元数値データ３－１は１データ単位の構成を採る。各単位データはまずｚ方向３－２－１に並び、その列がｘ方向３－２－２に並んでいる。さらに（ｘサイズ，ｙサイズ，ｚサイズ）＝（ｍ，１，ｎ）（ただし、ｍ，ｎ＝各方向の最大サイズ、１は数値である）で構成される面はｙ方向３－２－３に並んでいる。すなわち３次元数値データ３－１のデータ３－１－１～５を直線的な位置表現３－３で表すと、それぞれ矢印で示した対応となり３－３－１～５の位置となる。図３のデータ構成の場合のデータ構成識別子２－６を１と定義する。データの構成には他に、同じ１データ単位でｘ，ｙ，ｚ方向のデータの並びの優先度が違う構成や、（ｘサイズ，ｙサイズ，ｚサイズ）＝（ａ，ａ，１）のｓｑｕａｒｅ型を単位構成として並ぶ構成などがあり、それらにもそれぞれ２以上の一意な正数値を定義する。

【００１７】図４は図３の構成における１ボクセルのボリュームレンダリング計算処理での３次元角度（０°，０°，０°）（正面から見た場合である）での参照パターン例である。処理対象データの計算では、任意の角度に対しても精密なデータを得るために、対象データの近房のデータを用いて正規化する。本実施例では例えば対象データを４－１とした時、そのデータを含む周囲８点のデータ４－１～８を参照して計算する方法を採用する。さらにボリュームレンダリングでは反射処理のために対象データの法線ベクトルを求める必要があり、これには４－２～５、４－９～１１の位置にあるデータを用いて計算する。それぞれの位置データに対しても任意角度に対応した精度を出すため、そのデータを含む周囲８点を参照して求める。従って、データ４－１を対象とした計算処理において参照するデータとその位置を識別すると、図４に示した３２点となり、そのすべての点を含むデータ構成はｘ，ｙ，ｚ方向サイズがそれぞれ４の立方体構成となる。この値は１回目の計算で参照したデータの座標から各方向の最大値と最少値を求めることにより算出できる。１ボクセルを求める計算ではこの３２点のうち８点は２回以上参照しており、一度読み込んだこれらのデータをより高速にアクセスできる場所に保管し

ておければ２回目以降の参照は高速に行なえることから、容量は小さいがアクセス性能が速いローカルメモリにこのデータを保管するバッファを確保する。このバッファの最少サイズは前述で求めた構成の６４でよいが、ローカルメモリに余裕がある場合には、さらに効率の良いサイズを確保する。

【００１８】図５は求める画素（ピクセル）と参照するデータ群との一番単純な関係を示したものである。レイキャスティング方式では、画素（ピクセル）５－１～４はそれぞれデータ群５－５～８を図４に示した参照パターンで参照して計算する。すなわちｚ方向に並ぶデータを対象に計算処理が進む。例えば、図４でデータ４－１の次に計算対象となるのはデータ４－５である。データ４－１と４－５の計算で参照するデータは図４の４＊４＊４の立方体のデータ構成の中に２８個ある。次に計算対象になるのがデータ４－１２の場合は２０個ある。ｚ方向に１つサイズを大きくした直方体分のデータを読み込めるバッファがあれば、２４個になる。従ってｚ方向に並ぶデータをより多くバッファに格納できれば、再利用できるデータが多くなり効率がよい。図３に示した３次元数値データはｚ方向に並んでいること、共有メモリからデータを読み込む処理では１回に複数バイト読み込まれることを利用し、共有メモリから１回に読み込むバイト数の倍数をｚ方向のデータサイズに採用する。例えば１回に読み込むバイト数が１６の場合はｚ方向サイズ２－２０は１６の倍数とする。

【００１９】図６は、確保したバッファに共有メモリのどのデータが入っているかを管理するためのバッファ管理情報テーブルである。バッファの位置を識別するインデックス値６－１と現在そのバッファ位置に入っている３次元数値データの共有メモリ内でのアドレスを識別できる値６－２から成る。例えば、１６バイトづつデータがバッファに書き込まれるとすると、ｉｎｄｅｘ１には最初の１６バイトのデータが書き込まれるバッファ内のアドレスが書き込まれ、６－２には、その１６バイトデータの共有メモリ内のアドレスが書き込まれ、ｉｎｄｅｘ２には次の１６バイトのデータが書き込まれるバッファ内のアドレスが書き込まれ、６－２には、その１６バイトデータの共有メモリ内のアドレスが書き込まれ、以下同様に行われる。このバッファ管理情報テーブルについては図示されていないが、各プロセッサのローカルメモリに用意される。共有メモリアドレス識別子６－２の初期値はすべて－１である。バッファに１６バイト単位で格納する場合にはインデックスは（バッファ相対アドレス÷１６）で算出する。

【００２０】図７は確保したバッファ７－２と共有メモリ内の３次元数値データ７－１との対応を示したものである。共有メモリ内の３次元数値データ７－１の任意のｎバイト（ｎはバッファのｚ方向サイズ。例えば１６）のデータ群７－１－１はバッファ７－２内のエリア７－２－１に格納する。同様に共有メモリ内の３次元数値データ群７－１－２～７はバッファ７－２内のエリア７－２－２～５に格納する。この対応には対象となる３次元数値データの３次元数値データ先頭からの相対アドレスをバッファサイズで割り、その余りをバッファの相対アドレスとして用いている。従ってこの方法では、共有メモリ内の３次元数値データ群７－１－１と７－１－５がバッファのエリア７－２－１に格納され、３次元数値データ群７－１－３と７－１－７がバッファのエリア７－２－４に格納されるように、同じ位置に格納されることがある。

【００２１】そこで、この制御方式でも参照データが重なりあうことなく格納できるバッファサイズを求める処理を提供する。この計算には前述までの説明で求めた参照データの構成に基づくバッファサイズを用いる。図４に示したように隣あうｘ座標値やｙ座標値のデータ（例えば４－１と４－２）は参照データの開始／終了のｚ位置が近いことが多い。従って、バッファに読み込む時には、同じ計算で参照するデータがバッファ内で上書きしあわないように、まずバッファサイズはｚ方向サイズの公約数やｚ方向とｘ方向のサイズを掛け合わせた値の公約数を避けた値にする。例えば、もし公約数であれば、バッファのｘ，ｙ方向サイズを現在の値よりも大きくかつ一番小さな素数値に変更する。３次元数値データの各方向サイズを２５６とした場合、前述で求めた参照配置情報のｘ，ｚの方向サイズ４は公約数であるので、バッファのｘ，ｙ方向のサイズを現在の値よりも大きくかつ一番小さな素数５に変更する。これを図２のバッファのｘ，ｙ，ｚ方向サイズ２－１８～２０の初期値として登録する。これでｘ方向に並ぶデータが同じバッファ位置に格納されることはない。さらにｙ方向に並ぶデータにも対処できるように、バッファのｘ，ｙ，ｚ方向サイズを使って値を求めて加算し、バッファサイズを算出する。この値はバッファサイズ２－１７に算出したサイズを格納する。

【００２２】図８は算出したバッファサイズに基づくデータの格納結果である。本例では前述で求めたバッファのｘ，ｙ，ｚ方向サイズを掛け合わせた値に、ｘ，ｙ方向サイズを掛け合わせた値から１を引いた値にｚ方向サイズを掛けた値を加算して求める。式を以下に示す。

（ｘ＿Ｓｉｚｅ＊ｙ＿Ｓｉｚｅ＊ｚ＿Ｓｉｚｅ）＋（（ｘ＿Ｓｉｚｅ＊ｙ＿Ｓｉｚｅ－１）＊ｚ＿Ｓｉｚｅ）

上記の２項目の式で求めた値を加算することで、参照データが重なることなく格納できるバッファサイズが得られる。この式にｘ＿Ｓｉｚｅ＝５，ｙ＿Ｓｉｚｅ＝５，ｚ＿Ｓｉｚｅ＝１６を代入し７８４という値を算出する。このバッファサイズで図６～７で示した格納操作をする。

【００２３】図８は、図３に示す３次元数値データの

ｘ，ｙ，ｚの各方向サイズを２５６とし、３次元数値データ３－１－１から始まり、ｚ方向を１６、ｘ方向を５、ｙ方向を５とした直方体内のデータをバッファに書き込み格納した場合のバッファ内におけるデータの配置状況を示したものである。この場合、図３において、３－１－１～３－１－４の３次元数値データは１番上の（ｙ＝０の）ｘｚ面のデータであり、３－１－５の３次元数値データは２番目の（ｙ＝１の）ｘｚ面のデータであり、同様に３－１－５の１段下の３次元数値データは３番目の（ｙ＝２の）ｘｚ面のデータであり、以下同様である。データが１番上の（ｙ＝０の）ｘｚ面に属することをｙ方向が「256^2*0」であると表わし、データが２番目の（ｙ＝１の）ｘｚ面に属することをｙ方向が「256^2*1」であると表わし、以下同様に、256^2*2、256^2*3、256^2*4で表わす。また、３－１－１から始まるｙ＝０での１６個のデータ、３－１－５からから始まるｙ＝１での１６個のデータ、・・・ｙ＝４での１６個のデータをｘｚ方向が０～１５（番地）のデータとして表わし、３－１－３から始まるｙ＝０での１６個のデータ、・・・ｙ＝４での１６個のデータをｘｚ方向が２５６～２７１（番地）のデータとして表わし、３－１－３の右隣の位置から始まるｙ＝０での１６個のデータ、・・・ｙ＝４での１６個のデータをｘｚ方向が５１２～５２７（番地）のデータとして表わし、　さらにその右隣の位置から始まるｙ＝０での１６個のデータ、・・・ｙ＝４での１６個のデータをｘｚ方向が７６８～７８３（番地）のデータとして表わし、さらにその右隣の位置から始まるｙ＝０での１６個のデータ、・・・ｙ＝４での１６個のデータをｘｚ方向が１０２４～１０２８（番地）のデータとして表わす。そして、図８において、ｘｚ方向が０～１５で、ｙ方向が256^2*0である欄は３次元数値データの０～１５（３－１－１から始まる１６個のデータ）に対応しており、この欄には、３次元数値データの０～１５がバッファに格納された場合のそのバッファ内での格納位置情報が書き込まれる。このバッファ内での格納位置（アドレス）は、データの共有メモリアドレスの値をバッファサイズの値で割った余りの値である。上記の場合、データの共有メモリアドレスの値は０～１５であるから、これをバッファサイズの値７８４で割った余りの値は０～１５であり、この値０～１５が、図８の表のｘｚ方向が０～１５で、ｙ方向が256^2*0である欄に記入されている。また、３－１－３で始まる１６個の３次元数値データ場合は、そのデータの共有メモリアドレスの値は２５６～２７１であるから、この値をバッファサイズの値７８４で割った余りの値は２５６～２７１であり、この値２５６～２７１がバッファ内での格納位置（アドレス）であり、この値が図８の表のｘｚ方向が２５６～２７１で、ｙ方向が256^2*0である欄に記入されている。さらに、３－１－５で始まる１６個の３次元数値データ場合は、そのデータの共有メモリアドレスの値は６５５３６～６５５５１であるから、この値をバッファサイズの値７８４で割った余りの値は４６４～４７９であり、この値４６４～４７９がバッファ内での格納位置（アドレス）であり、この値が図８の表のｘｚ方向が０～１５で、ｙ方向が256^2*1である欄に記入されている。

【００２４】この結果からすべて異なるバッファ位置に格納されていることが判る。なお、（Ｓｉｚｅ＿ｘ，Ｓｉｚｅ＿ｙ，Ｓｉｚｅ＿ｚ）＝（５、５、３２）、（５、５、４８）、（７、７、１６）なども同様にすべて異なるバッファ位置に格納されることは確認済みである。また、ｚ＞０の場合でも隣あうデータとの距離は同じなので同様の結果が得られる。上記までの手段で求めたバッファ構成とサイズを持つバッファをローカルメモリに確保することにより、例えば、角度（０°，０°，０°）において２５回の共有メモリのリードアクセスによって、７８４バイトのサイズのバッファの中に１３ボクセル分の計算で参照するデータ１７４個をローカルメモリに持つことができる。これは６２４回の参照分のデータであり、共有メモリよりローカルメモリが５倍速いとすると約４倍の高速化が図れる。

【００２５】図９は、ルートプロセッサが持つ全プロセッサの稼働状況管理テーブル１－４－３である。管理テーブル９－１は、接続されているプロセッサの数までの値をとるインデックス部９－２と、稼働可能の通知を通信してきたプロセッサ上の描画プロセスの一意な識別子を格納する部分９－３から成る。ルートプロセッサは接続されているプロセッサの数を得てこの管理テーブルを作成する。識別子部分９－３の初期値は－１が入っている。各プロセッサはローカルメモリが確保でき稼働可能になったならばルートプロセッサに稼働可能の通知を送り、ローカルメモリが確保できない場合は稼働不可能の通知を送る。ルートプロセッサは送られて来た内容を判定し、稼働可能の通知を送ってきたプロセスの識別子を管理テーブルのプロセス識別子９－３に格納する。接続されている全プロセッサの通知を受けた後、稼働不可能の通知を送ってきたプロセッサが行うはずだった処理を稼働可能なプロセッサ数で再分割し、稼働可能なプロセスの識別子を用いて配送する。

【００２６】図１０は、各プロセッサでの処理において、対象３次元数値データを格納するバッファをローカルメモリに確保する処理の流れを示したものである。ユーザからの要求があると描画処理であるかを判定し（１０－１）、描画処理でないならば対応する処理を実行する（１０－２）。描画処理ならばルートプロセッサは処理の分割を行ない（１０－３）、分担処理の情報・描画条件情報・３次元数値データの全体サイズ・構成情報等を付加した描画要求を各プロセッサに渡し（１０－４）、各プロセッサは要求を受け取る（１０－５）。次に管理情報テーブルのバッファサイズ２－１７が０かを

判定し（１０－６）、０であれば１回目のボクセル計算でバッファのサイズを算出する（１０－７）。ローカルメモリの空き領域サイズを入手し（１０－８）、バッファサイズを算出してバッファを確保する（１０－９）。その後確保したバッファを用いて描画処理を行なう（１０－１０）。

【００２７】図１１は、バッファの使用制御の処理の流れである。各プロセッサは割り当てられた画素を求めるのに参照するデータの共有メモリアドレスを得て（１１－１）、その値をバッファサイズで割った余りでバッファのどの位置に格納するかを求める（１１－１２）。バッファ管理情報テーブル（図６）を参照し該当位置にすでにそのデータがあるかを判定し（１１－３）、あればバッファからその内容を参照し（１１－４）、なければ共有メモリから読み込んで参照する（１１－５）。読み込んだデータは、事前に算出したバッファの位置に格納し（１１－６）、アドレスを管理テーブルに格納する（１１－７）。同じインデックス値を持つデータが来た場合には、重ね書きをする。

【００２８】図１２は、描画計算処理中に、計算に必要なローカルメモリが確保できない場合のプロセッサにおける処理の流れである。バッファサイズ削減の有無を識別する削減フラグを用意し０で初期化する（１２－１）。ローカルメモリの空き領域を求め（１２－２）、現在の指定されたバッファサイズでバッファが確保できるかを判断し（１２－３）、可能であれば確保して（１２－４）、描画処理を実行する（１２－５）。現在のバッファサイズで確保できない場合はローカルメモリの空き領域サイズと規定した最少バッファサイズとを比較し（１２－６）、最少バッファサイズよりも大きい場合は最少バッファサイズと空き領域サイズとの間の値でバッファサイズを求めバッファを確保する（１２－７）。この時求めたバッファのｘ，ｙ，ｚ方向サイズは管理情報テーブルに再登録し（１２－８）、削減フラグに１を代入する（１２－９）。得たバッファを使って描画処理を実行する（１２－５）。ローカルメモリの空き領域サイズが最少バッファよりも小さい場合には処理の分割を行なっているルートプロセッサにその旨を通知し（１２－１０）、処理を終了する。ルートプロセッサでは通知を受けると（１２－１１）、該当するプロセッサの担当処理を稼働可能なプロセッサ数で分割し（１２－１２）、稼働可能なプロセッサに再分配する（１２－１３）。ルートプロセッサも処理を分担し描画処理を行なう（１２－１４）。他のプロセッサは通知を受け取り（１２－１５）、描画処理を行なう（１２－１６）。この処理により、いずれかのプロセッサでローカルメモリを確保できなくても処理を続けることができる。

【００２９】図１３はバッファのｚ方向サイズの再算出処理の流れを示したものである。図１０の（１０－１０）にある描画処理中に、１ピクセルの処理で参照するデータのｚ方向の最大サイズ（レイ・キャスティングにおける視線トレース回数）を算出する（１３－１）。算出したｚ方向最大サイズがバッファのｚサイズよりもデータ読み込み単位数以上に大きいかを判断し（１３－２）、大きい場合には次に削減フラグが０でメモリに余裕があるかを判断し（１３－３）、０であれば描画処理終了後にバッファのｚ方向のサイズを算出した値よりも大きくかつ一番近いデータ読み込み単位数の倍数に置き換える（１３－４）。削減フラグが１の場合には何もしない。また、算出したｚ方向最大サイズがバッファのｚサイズよりもデータ読み込み単位数以上に小さい場合にも、描画処理終了後にバッファのｚ方向のサイズを算出した値よりも大きくかつ一番近いデータ読み込み単位数の倍数に置き換える（１３－４）。この値は次回の計算時に最適なバッファサイズを算出するために使用する。

【００３０】図１４は本発明の第２実施例であり、本発明を適用した別システムの例である。本実施例は第１実施例で示した構成をサブシステムとし、ホストプロセッサ１４－１と接続する構成とする。ホストプロセッサ１４－１には描画処理を分割し各プロセッサに分配・通信する処理１４－１－１がある。また、ホスト側のメモリ１４－２には接続されているプロセッサの稼働状況情報１４－２－１がある。ホスト側は分配・通信処理１４－１－１で処理をプロセッサ１－１～３に分配し、各プロセッサ１－１～３はその内容を受信して描画処理を行なう。描画処理とバッファに関する処理は第１の実施例と同じである。但し第１の実施例と異なり、ローカルメモリにバッファが確保できない場合には、ホストプロセッサ１４－１にその旨の通知を送る。各プロセッサ１－１～３はその内部にこの手段１－１－３、１－２－３、１－３－３を持つ。ホストプロセッサ１４－１はこの通知を受信し稼働可能な接続プロセッサの数で処理を再分割し稼働可能なプロセッサに分配送信する手段１４－１－２も持つ。この手段４－１　２を用いて処理を再分配する。各プロセッサ１－１～３は再分配された処理を受信する。受信後の処理は第１実施例と同じである。

【００３１】

【発明の効果】本発明は参照するデータのみローカルメモリに読み込む方式のため、よけいなデータ転送オーバヘッドがない。またすべての参照データを格納するバッファが確保できない場合でも、参照データの配置情報に適したサイズでバッファを確保するため、より小さなサイズで実現でき、かつデータの再参照時のヒット率も高くなり、再参照時のアクセス時間が短縮でき、高速な処理を実現できる。

【図面の簡単な説明】

【図１】本発明の実施例のシステムの構成を示す図である。

【図２】管理情報テーブルの例を示す図である。

【図３】３次元数値データの構成例を示す図である。

13

【図４】１ボクセルのボリュームレンダリング計算処理での３次元角度（０°，０°，０°）での参照パターン例を示す図である。

【図５】求める画素と参照するデータ群との単純な関係を示す図である。

【図６】バッファ管理情報テーブルの例を示す図である。

【図７】確保したバッファと共有メモリ内の３次元数値データとの対応を示す図である。

【図８】算出したバッファサイズに基づく３次元数値データのバッファへの格納結果を示す図である。

【図９】稼働状況管理テーブルの例を示す図である。

【図１０】バッファ確保処理のフローチャートを示す図である。

【図１１】バッファ使用制御のフローチャートを示す図である。

【図１２】ローカルメモリが確保できない場合の処理のフローチャートを示す図である。

14

【図１３】バッファのｚ方向サイズ再算出のフローチャートを示す図である。

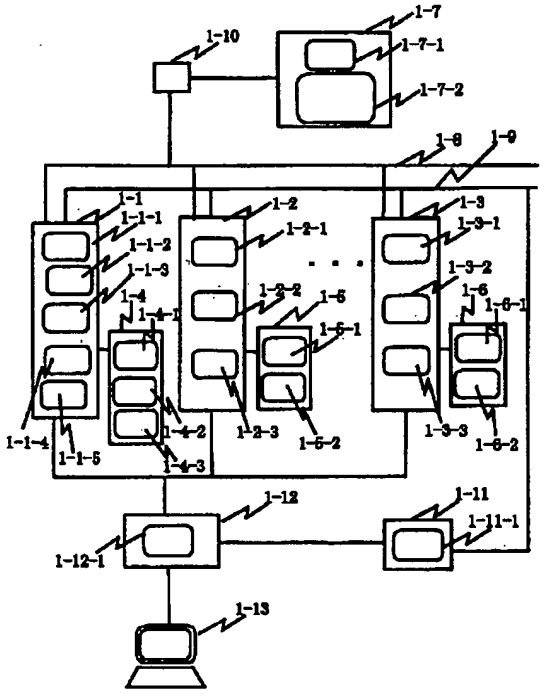
【図１４】第２の実施例のシステム構成を示す図である。

【符号の説明】

１－１　ルートプロセッサ
１－２～３　処理プロセッサ
１－４～６　ローカルメモリ
１－７　共有メモリ
１－８～９　バス
１－１０　バスブリッジ
１－１１　フレームバッファ
１－１２　表示プロセッサ
１－１３　表示装置
１４－１　ホストプロセッサ
１４－２　ホスト側のメモリ

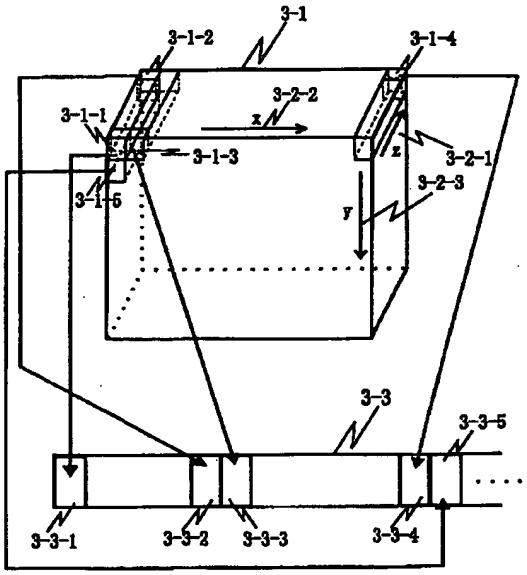
【図１】

図 1



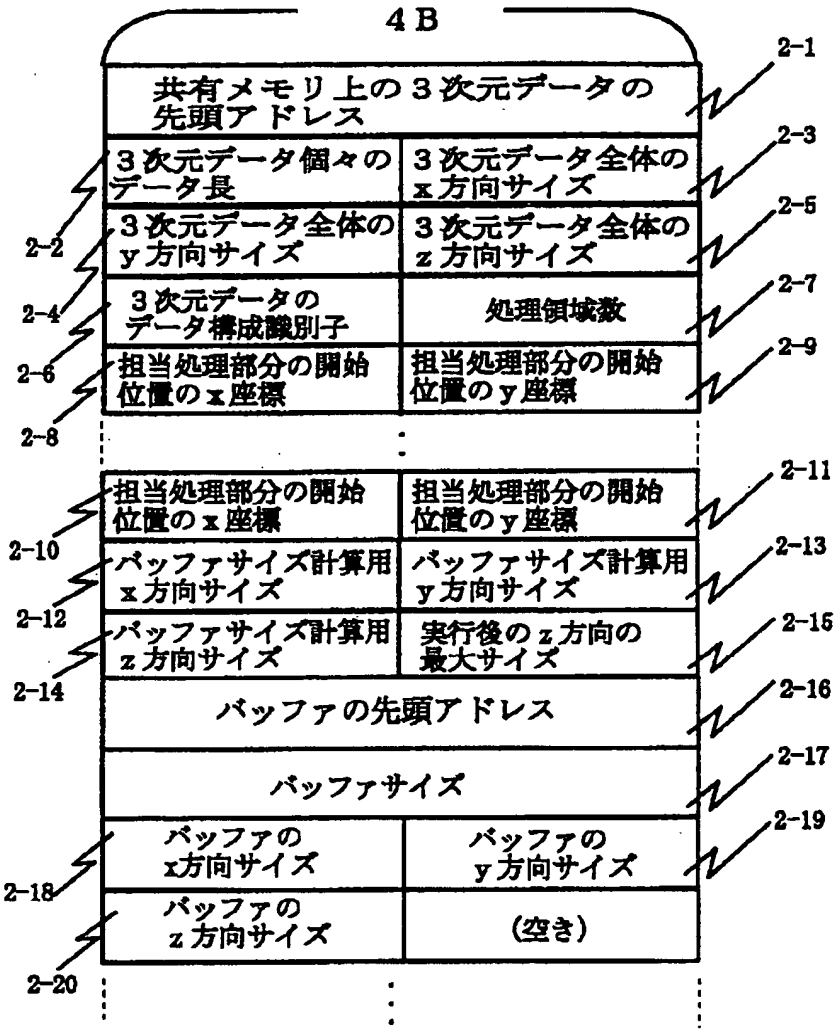
【図３】

図 3



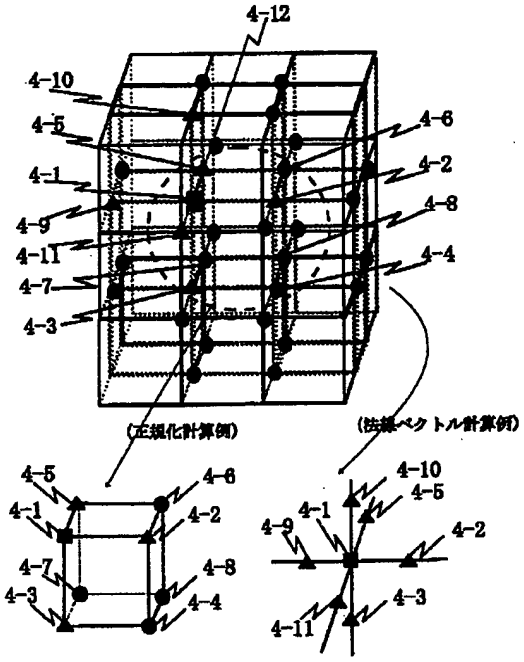
【図２】

図 2


4B
共有メモリ上の３次元データの先頭アドレス
2-1
３次元データ個々のデータ長
2-2
３次元データ全体のｘ方向サイズ
2-3
３次元データ全体のｙ方向サイズ
2-4
３次元データ全体のｚ方向サイズ
2-5
３次元データのデータ構成識別子
2-6
処理領域数
2-7
担当処理部分の開始位置のｘ座標
2-8
担当処理部分の開始位置のｙ座標
2-9
担当処理部分の開始位置のｘ座標
2-10
担当処理部分の開始位置のｙ座標
2-11
バッファサイズ計算用ｘ方向サイズ
2-12
バッファサイズ計算用ｙ方向サイズ
2-13
バッファサイズ計算用ｚ方向サイズ
2-14
実行後のｚ方向の最大サイズ
2-15
バッファの先頭アドレス
2-16
バッファサイズ
2-17
バッファのｘ方向サイズ
2-18
バッファのｙ方向サイズ
2-19
バッファのｚ方向サイズ
2-20
(空き)

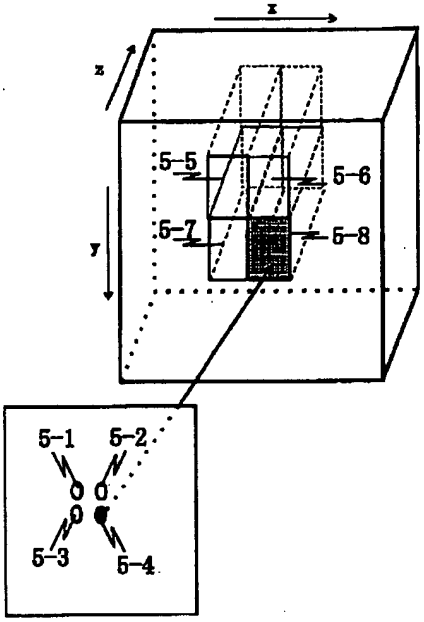
【図４】

図 4



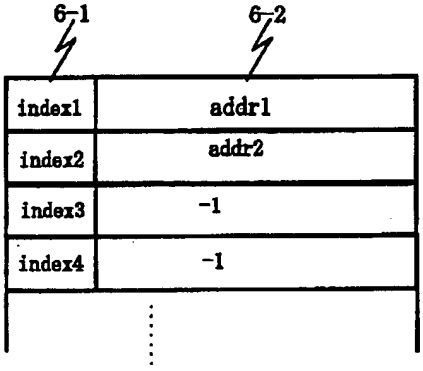
【図５】

図 5



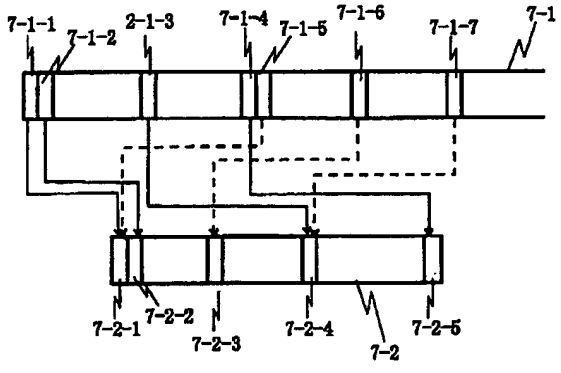
【図６】

図 6



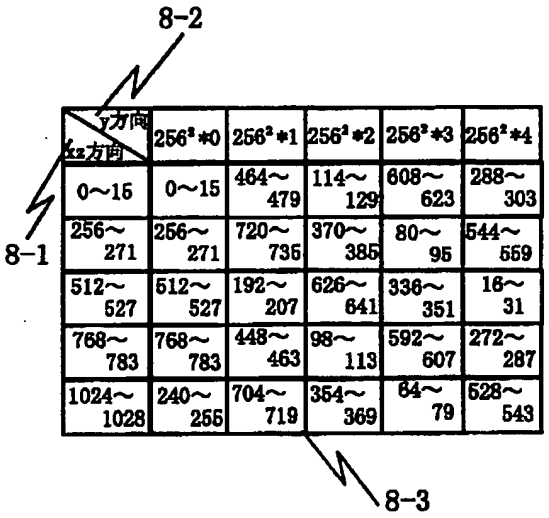
【図７】

図 7



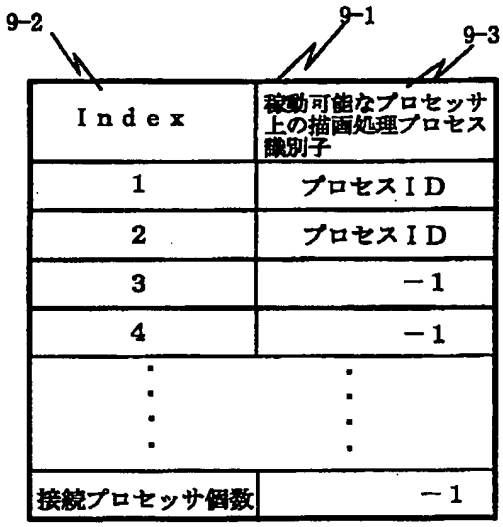
【図８】

図 8



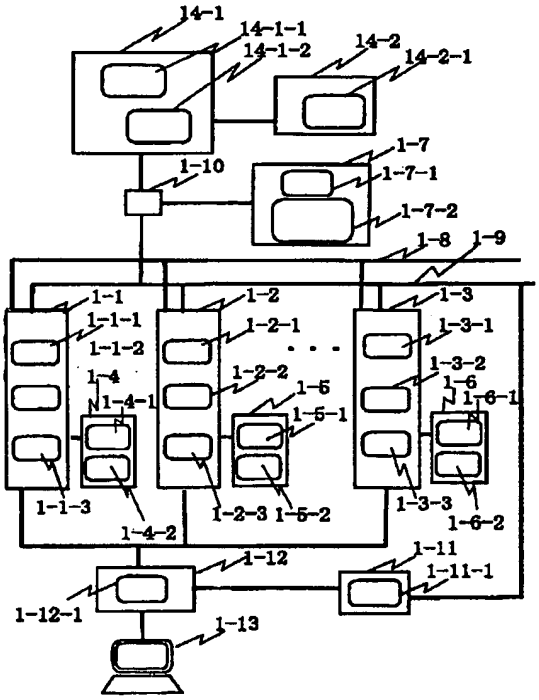
【図９】

図 9



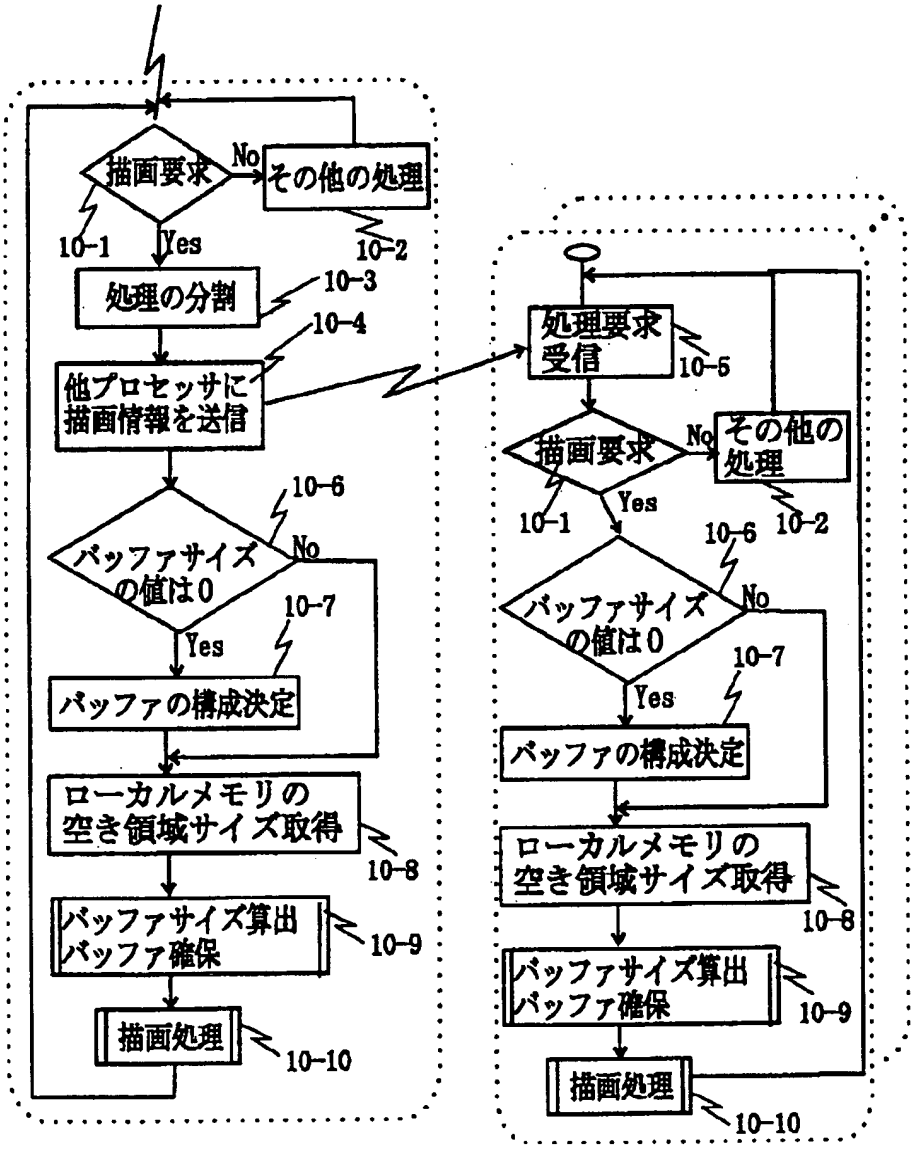
【図１４】

図 14



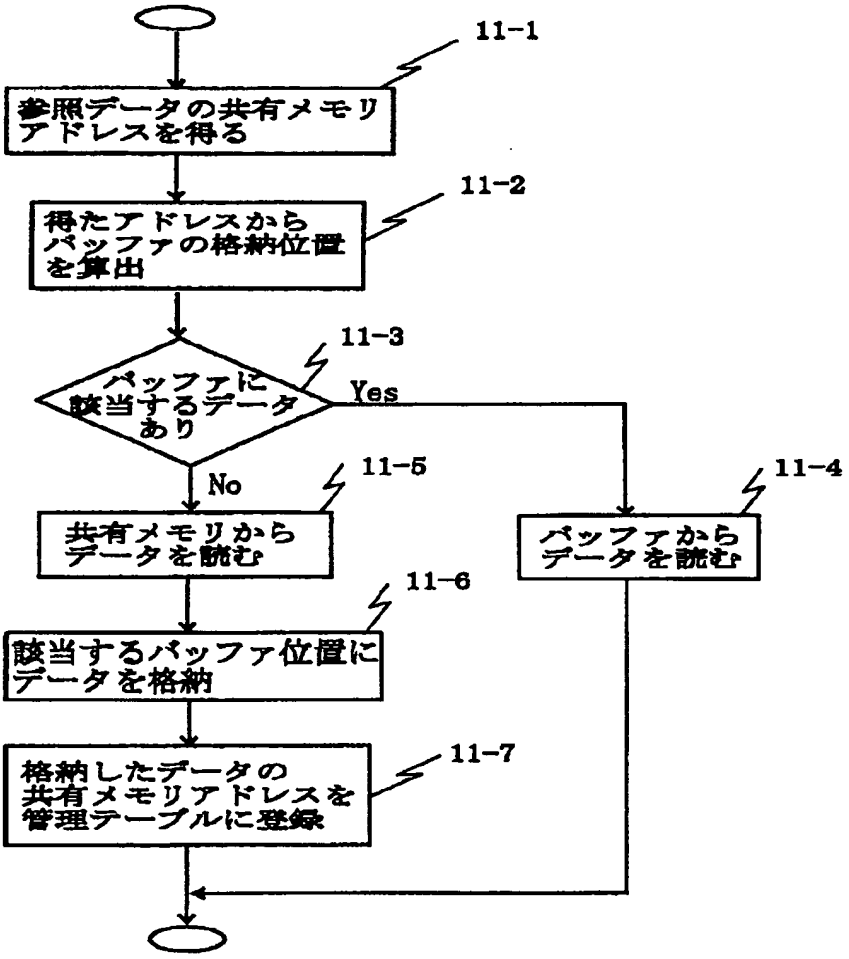
【図１０】

図 10


描画要求
No
その他の処理
10-1
Yes
10-2
処理の分割
10-3
10-4
他プロセッサに描画情報を送信
10-6
バッファサイズの値は0
No
Yes
10-7
バッファの構成決定
ローカルメモリの空き領域サイズ取得
10-8
バッファサイズ算出 バッファ確保
10-9
描画処理
10-10
処理要求受信
10-5
描画要求
No
その他の処理
10-1
Yes
10-2
10-6
バッファサイズの値は0
No
Yes
10-7
バッファの構成決定
ローカルメモリの空き領域サイズ取得
10-8
バッファサイズ算出 バッファ確保
10-9
描画処理
10-10

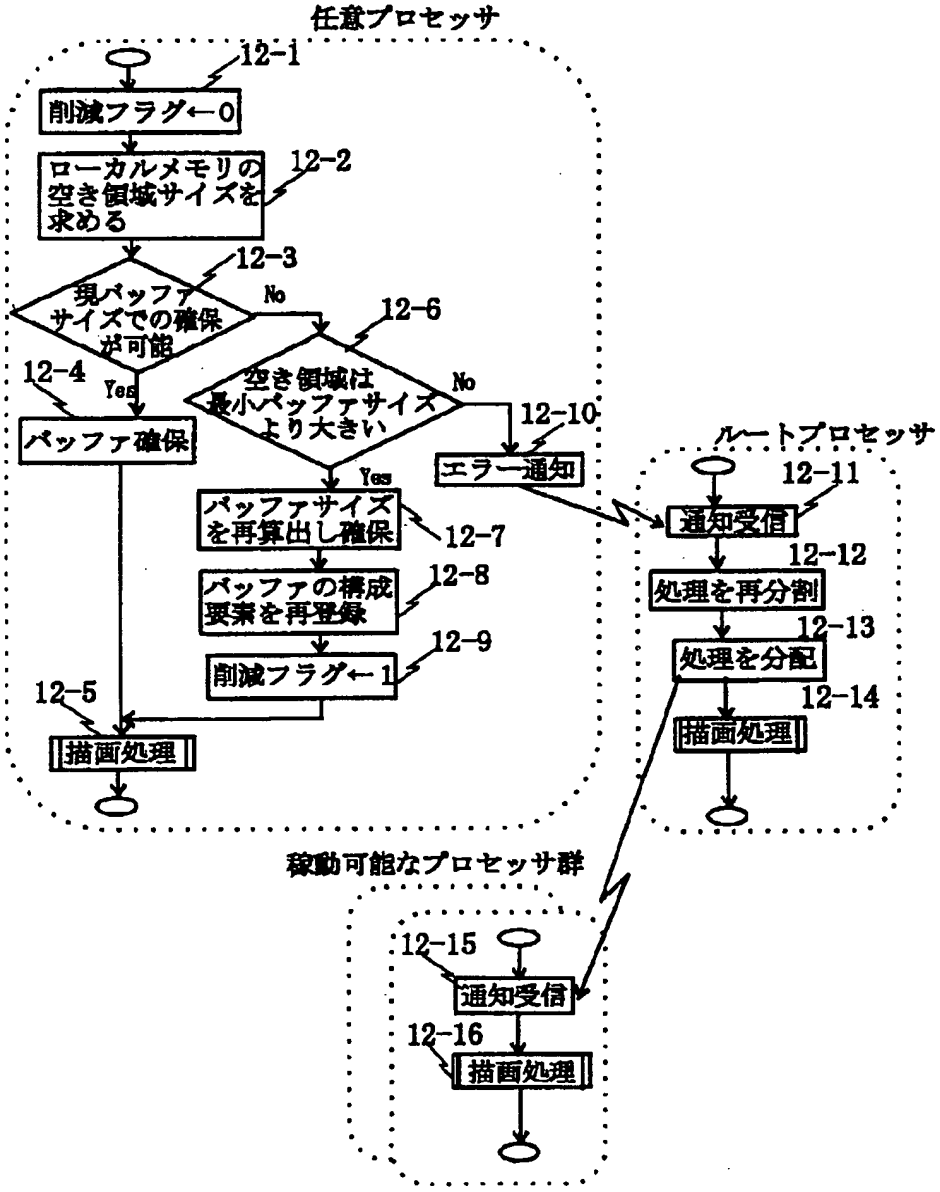
【図１１】

図 11


11-1
参照データの共有メモリアドレスを得る
11-2
得たアドレスからバッファの格納位置を算出
11-3
バッファに該当するデータあり
Yes
No
11-5
共有メモリからデータを読む
11-4
バッファからデータを読む
11-6
該当するバッファ位置にデータを格納
11-7
格納したデータの共有メモリアドレスを管理テーブルに登録

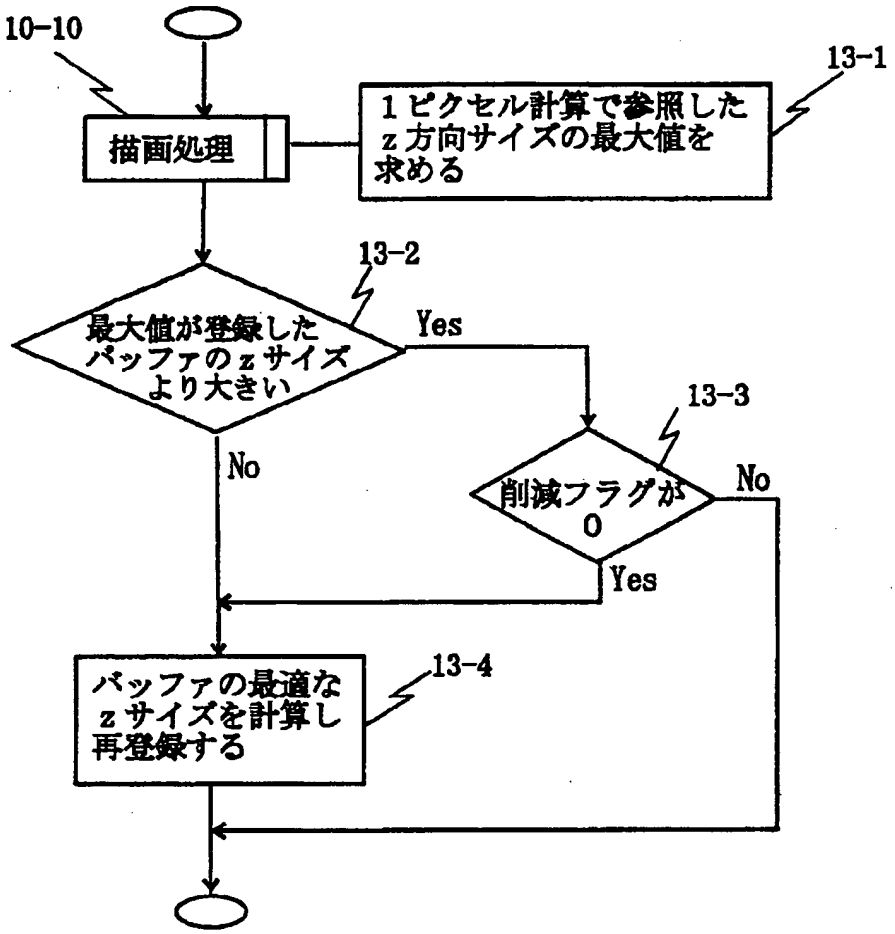
【図１２】

図 12


任意プロセッサ
12-1
削減フラグ←0
ローカルメモリの空き領域サイズを求める
12-2
12-3
現バッファサイズでの確保が可能
No
12-6
空き領域は最小バッファサイズより大きい
No
12-10
エラー通知
12-4
Yes
バッファ確保
Yes
バッファサイズを再算出し確保
12-7
バッファの構成要素を再登録
12-8
削減フラグ←1
12-9
12-5
描画処理
ルートプロセッサ
12-11
通知受信
12-12
処理を再分割
12-13
処理を分配
12-14
描画処理
稼動可能なプロセッサ群
12-15
通知受信
12-16
描画処理

【図１３】

図 13


10-10
描画処理
13-1
１ピクセル計算で参照した
ｚ方向サイズの最大値を
求める
13-2
最大値が登録した
バッファのｚサイズ
より大きい
Yes
No
13-3
削減フラグが
0
No
Yes
13-4
バッファの最適な
ｚサイズを計算し
再登録する